

AFRL-IF-RS-TR-2002-280
Final Technical Report
October 2002



INFORMATION TOOLS FOR SECURITY PROTECTION

Odyssey Research Associates, Incorporated

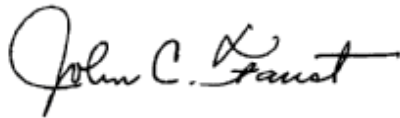
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

**AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK**

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2002-280 has been reviewed and is approved for publication

APPROVED:

A handwritten signature in black ink, appearing to read "John C. Faust". The signature is fluid and cursive, with a long horizontal stroke at the end.

JOHN FAUST
Project Engineer

FOR THE DIRECTOR:

A handwritten signature in black ink, appearing to read "Warren H. Debany". The signature is fluid and cursive, with a long horizontal stroke at the end.

WARREN H. DEBANY, Technical Advisor
Information Grid Division
Information Directorate

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 074-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE October 2002	3. REPORT TYPE AND DATES COVERED Final Apr 96 – Aug 97		
4. TITLE AND SUBTITLE INFORMATION TOOLS FOR SECURITY PROTECTION		5. FUNDING NUMBERS C - F30602-96-C-0091 PE - 33140F PR - 7820 TA - 04 WU - 29		
6. AUTHOR(S) David Rosenthal, Peter Samsel, and Cheryl Barbasch				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Odyssey Research Associates, Incorporated Cornell Business & Technology Park 33 Thornwood Drive, Suite 500 Ithaca New York 14850-1250		8. PERFORMING ORGANIZATION REPORT NUMBER		
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory/IFGB 525 Brooks Road Rome New York 13441-4505		10. SPONSORING / MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TR-2002-280		
11. SUPPLEMENTARY NOTES AFRL Project Engineer: John C. Faust/IFGB/(315) 330-4544/ John.Faust@rl.af.mil				
12a. DISTRIBUTION / AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 Words) This report describes the three tools built for the Information Security Tools project. Each of these tools provides a way of helping a user uncover potential security problems arising from the composition of parts of a system. The intent of this work was to provide tools usable by trained, but not necessarily expert, security personnel. The System Security Analysis Tool helps a user to model and analyze the security of systems. It provides a way to combine a security analysis of a generic architecture with the specific requirements of a particular system. The Application System Call Analysis Tool is used to identify a trusted application's system calls, and display potential problems or limitations associated with those calls. It is intended for use with an HP-CMW. The Application Installation Analysis Tool is used to check the access-control policy of a trusted application on an HP-CMW using a vendor's abstract policy, installation parameters, and the current state of the system. It provides a way of identifying differences between the way an access control policy is implemented and the vendor's policy.				
14. SUBJECT TERMS Information Security, System Security Analysis, Composition, Application System Call Analysis, Application Installation Analysis			15. NUMBER OF PAGES 25	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

Table of Contents

List of Figures	iii
1 Summary.	1
2 Introduction	1
3 System Security Analysis Tool	2
3.1 Creating or modifying a system design.....	2
3.2 Incorporation of Generic Components	4
3.3 Security tests	5
3.4 System Security Analysis Tool Example	6
4 Application System Call Analysis Tool	6
5 Application Installation Analysis Tool.....	7
5.1 Security Specification File format.....	7
5.2 <i>InstallCheck</i> example	8
6 Conclusions	8
7 Recommendations	8
8 References	9
9 Appendix A: Example Report of System Security Analysis Tool.....	10
9.1 Introduction	10
9.2 System Overview	10
9.3 Component Summary.....	10
9.3.1 Satellite.....	10
9.3.2 HighEnclave1	10
9.3.3 HighEnclave2.....	11
9.3.4 IsseGuard.....	11
9.3.5 LowLAN	11
9.4 Component Security Summary	11
9.4.1 Satellite.....	11
9.4.2 HighEnclave1	11
9.4.3 HighEnclave2.....	11
9.4.4 IsseGuard.....	11
9.4.5 LowLAN	11

9.5	Link Security Summary	11
9.5.1	High1toHigh2.....	11
9.5.2	High1toIsseGuard.....	11
9.5.3	IsseGuardtoLowLAN.....	12
9.5.4	SatellitetoHigh1.....	12
9.6	Security Analysis Tests	12
9.6.1	ComponentLinkLevels [Potential Problem Detected]	12
9.6.2	GenericToSpecific [Potential Problem Detected]	12
9.6.3	PhysicalProtection [Potential Problem Detected]	12
9.7	Component Results	12
9.7.1	Satellite.....	12
9.7.2	HighEnclave1	12
9.7.3	HighEnclave2	12
9.7.4	IsseGuard.....	12
9.7.5	LowLAN	12
9.8	Generic Security Summary: IsseGuardedSystem	12
9.8.1	Generic Description.....	12
9.8.2	Generic Security Assessment	13
9.8.3	Generic Requirements and Specific Issues	13
10	Appendix B: Example of Application System Call Analysis Tool	14
10.1	ASCA description file: warnings.dsc	14
10.2	ASCA filter file: filter_file.flt	15
10.3	ASCA command-line execution.....	15
10.4	ASCA output: asca_out.....	15
11	Appendix C: Example of <i>InstallCheck</i>	16
12	List of Symbols	19

List of Figures

Figure 1	3
Figure 2	4
Figure 3	5
Figure 4	6
Figure 5	10

1 Summary

The purpose of this work was to develop security tools that could aid in handling security concerns of composite systems. The tools were designed so that individuals familiar with security, but who were not necessarily security experts, could use them. Each tool illustrates a different kind of analysis.

The three tools are:

- System Security Analysis Tool
- Application System Call Analysis Tool
- Application Installation Analysis tool

The System Security Analysis Tool is used to model and analyze the security of systems. It extends the NSA System Security Profiling methodology. The tool provides a way to combine a security analysis of a generic architecture with the specific requirements of a particular system. A generic architecture can be developed and analyzed prior to use by someone developing a specific system.

The Application System Call Analysis Tool identifies a trusted application's system calls and displays potential problems or limitations associated with those calls. It illustrates a way to connect vendor information, or security product profile information, with the security analysis of a trusted application.

The Application Installation Analysis tool is used to check the access-control policy of a trusted application on an HP-CMW using a vendor's abstract policy, installation parameters, and the current state of the system. It provides a way to identify differences between the way an access control policy is implemented on the system and the vendor's security policy.

In each case, the tools are designed to simplify the analysis by making use of existing security information about a part of the system.

2 Introduction

This report describes the three tools built for the Information Security Tools project. Each of these tools provides a way of helping a user uncover potential security problems arising from the composition of parts of a system. The intent of this work was to provide tools that are usable by trained, but not necessarily expert, security personnel.

These tools are:

- System Security Analysis Tool
- Application System Call Analysis Tool
- Application Installation Analysis tool (*InstallCheck*)

The System Security Analysis Tool helps a user to model and analyze the security of systems. The Application System Call Analysis Tool is used to identify system calls, and display potential problems or limitations associated with those calls. The Application Installation Analysis tool is used to check the access-control policy of a trusted application on an HP-CMW using a vendor's abstract policy, installation parameters, and the current state of a system. Note that these tools are independent of each other.

This effort builds on previous work performed at Odyssey Research Associates (ORA), specifically, the "Final Report on Composability Methodology." That effort provides material on the motivation of our approach and describes related work. The central concepts from the Composability work that are used in these tools are methods for factoring the security analysis. This report is self-contained, but limited to the description and purpose of the security tools.

For a complete description of the tools and how each one can be used, see the individual Software User Manual for that tool.

3 System Security Analysis Tool

The System Security Analysis Tool is used to aid in the security analysis and certification of systems. It provides a way of building an entity-attribute-relationship model of a system and its security characteristics. It is intended to be used in conjunction with Vitech's CORE (see CORE User Reference Manual) and it incorporates the existing NSA System Security Profiling structure.

The security tool can be used to:

- create or make changes to a system design
- incorporate information about generic components
- run certain security tests

3.1 Creating or modifying a system design

The tool supports creating and editing a system description. It is intended primarily for modifying and updating systems created with CORE. However, it can be used to build the elements, relationships, and attributes of system descriptions from scratch.

The main functionality of the system is accessible from the Browser. It can be used to:

- view relationships of system elements
- bring up other dialogs with more detailed element views for viewing and editing
- perform a security analysis
- invoke a dialog for incorporating information about generic components into a system description

The style of creation, deletion, and updates is similar, but not identical, to that of CORE. Using the browser, one can examine the elements that have been created and how those elements are related to each other (see Figure 1 below). For example, one can see that the component *HighEnclave1* was *builtin* system *SatImageSystem*. It's also possible to select particular elements from the browser and activate a window to view or update attributes (such as the element description) associated with that element. This is illustrated in Figure 2.

To aid in building or modifying a system, there are two special view windows which are accessible from the Browser. One dialog produces a picture of the subcomponents of a component and how they are linked, as well as security information associated with that component. The other dialog produces a hierarchical view of a component and its descendants based on a relationship.

Figure 1 shows the Browser dialog.

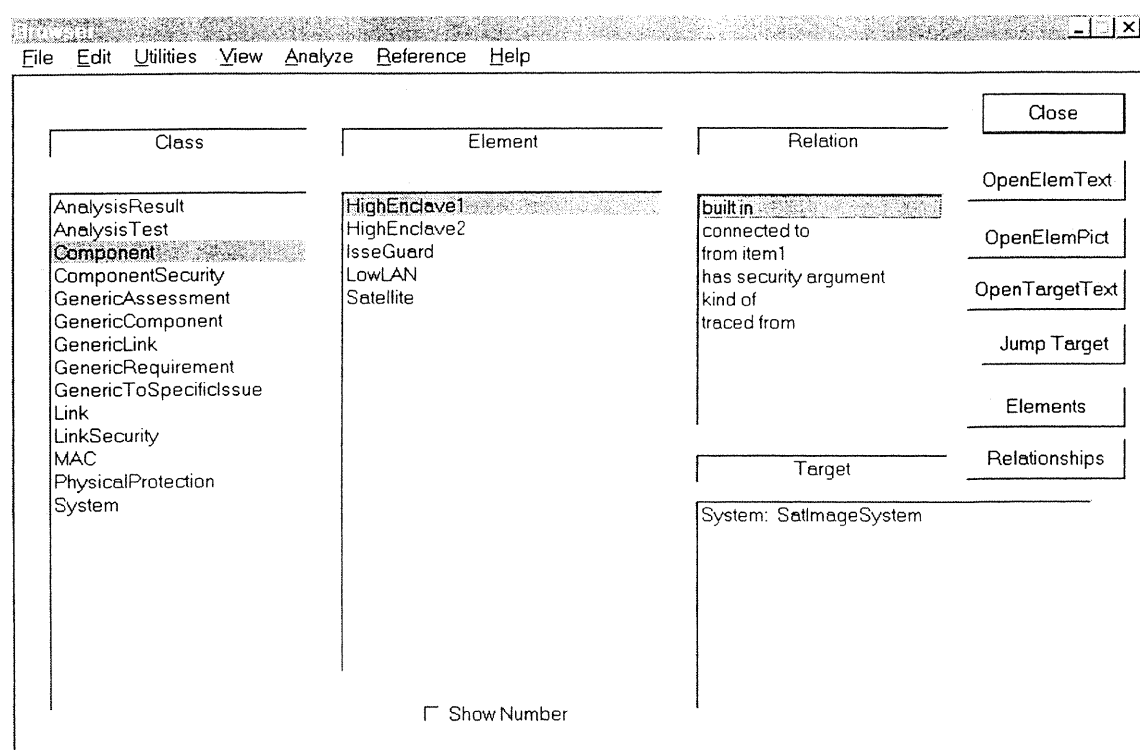


Figure 1

ISA - Image Component ISSEGuard

Close

Description

The ISSE Guard downgrades imagery from the high side Enclave1 to a low side LAN for dissemination.

Abbreviation

Created 5 May 1997 at 14:26:26

Creator System Engineer

Last Modified 8 May 1997 at 10:11:47

Number 1.4

Component Type nil

Relationships

Targets of Relationships

Open Target

built in
connected to
has security argument
kind of

System: SatImageSystem

Figure 2

The system descriptions (“rdt files”) of this tool are compatible with CORE. A system can be created or modified from either tool and used by the other.

3.2 Incorporation of Generic Components

One of the features of this tool is to facilitate relating components of a system with “generic” components from a library. The intent is that the security analysis of components can sometimes be factored into generic and specific parts. The generic part can be evaluated independently of a particular system, and then a system developer will only have to address residual issues that depend on the particular system configuration.

The system supports a tree list control to locate the desired component from the library (see Figure 3). All of the elements and relationships about the generic component can then be included with the system description.

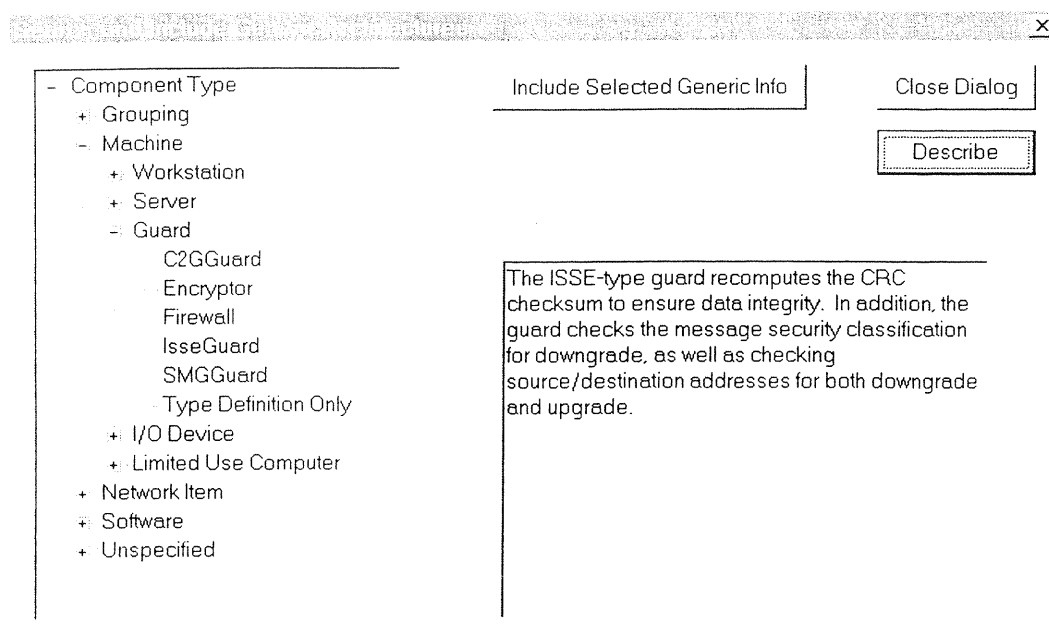


Figure 3

When creating new generic models, two attributes of generic components are particularly important. The first is the type of the generic component. For example, it might be a guard, operating system, or workstation. This information is used to place the model in its appropriate spot in the library. The other important attribute is the “library” attribute which, designates the main generic component. (A model might be composed of a number of generic components, and the tool uses the library attribute to determine which one of these to use for the tree list control.)

Of course, these models have a number of other attributes that need to be appropriately filled in. The description fields are particularly important.

A CORE report template has been provided that can be used to summarize the results of the security tests and describe how a specific system meets its generic requirements. It also contains some background material. This feature is only available to CORE users.

3.3 Security tests

The tool provides a number of security checks that can be applied to a system. A user can select the appropriate tests and decide whether to save the result in a file (see Figure 4). In this version of the system, there are three tests.

One test consists of a check on whether the MAC hierarchical levels associated with a link are compatible with the MAC hierarchical levels that a link is connected to. When

no MAC level is specified, the level is treated as UNCLASSIFIED. Currently, compartments are not checked.

Another test is to check on whether there is an explanation about physical protection associated with classified enclaves. Note that a component's generic type is determined by the "type" of the generic component it is built from.

The other test checks whether all generic requirements of a generic component generalizing a component are addressed by that component.

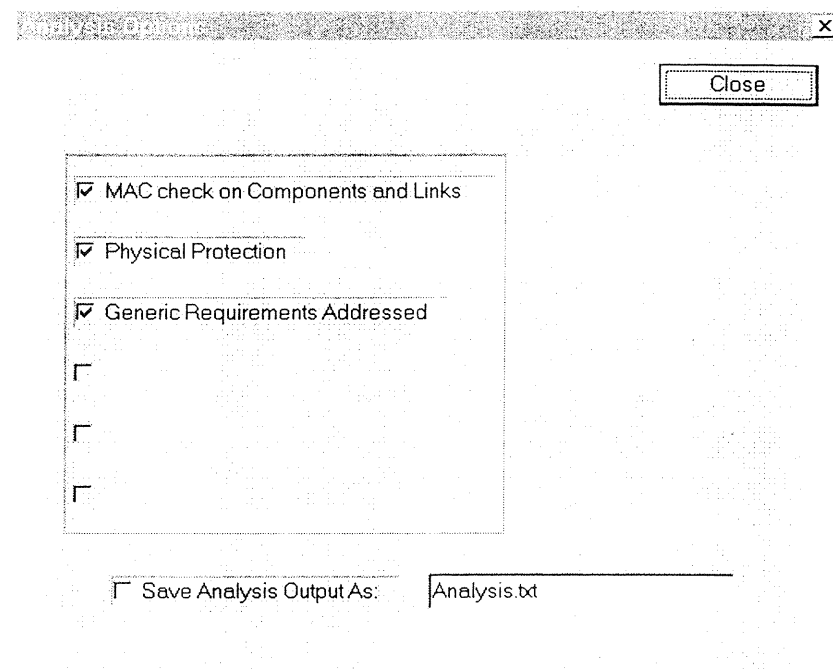


Figure 4

3.4 System Security Analysis Tool Example

Appendix A contains the result of running a CORE report script (with style modified) on a model which uses a guard. It summarizes key parts of the model, describes the generic components, and the security analysis results.

4 Application System Call Analysis Tool

The HP-CMW Application System Call Analysis (ASCA) Tool aids the user or system administrator in analyzing security implications of an application's system calls. In particular, the ASCA Tool reports the system calls used, and potential operational or security weaknesses associated with those calls. The ASCA Tool is a UNIX command-

line executable. It takes the application's archive, object, or non-stripped executable file(s) as input. The user may specify a system call filter file that prevents specific system calls from being displayed.

An example illustrating the tool is contained in Appendix B.

5 Application Installation Analysis Tool

The purpose of the Application Installation Analysis tool (*InstallCheck*) is to support the security analysis of a trusted application installation for the HP CMW. The tool helps the application developer specify the required security attributes associated with the application's subjects and objects, and then helps the installer of the application verify that the installation meets the specification.

This tool consists of a trusted UNIX command-line executable to be run by a user who possesses the *operator* or *isso* command authorization. The developer of the trusted application creates specification files for the security assumptions of the installation of the application. The installer specifies additional security parameters for the particular installation. Later, using the customized specification files, the security officer runs the *InstallCheck* tool to report discrepancies between the installed application and the specification.

5.1 Security Specification File format

The **Security Specification File** describes trusted users, objects, and access specifications for the trusted application. The purpose of the file is to document the security policy information at a higher level of abstraction than occurs in a File Attribute List.

The application vendor (developer) creates this file, which should not be modified by the user (installer). Any installation-specific parameters are saved as a parameter list that can be used for checking at a later time.

A class of users with common security privileges can be defined as a set of UNIX Ids, parameters that are replaced by Ids during installation, and other user classes. A UNIX group name may be associated with that class. Security attributes associated with this class are `BASE_PRIVILEGES`, `KERNEL_PRIVILEGES`, and `COMMAND_AUTHS`.

A class of files with common security privileges can be defined as the union of a set of UNIX files, parameterized filenames (which are defined during installation), and other file classes. Attributes, which may be defined for this class are: `is_executable`, `f_owner`, `f_group`, `f_mode`, `f_type`, `f_pprivs`, `f_gprivs`, `f_acl`, `f_slevel`, and `f_ilabel`. Partial specification constraints may also be made for these attributes.

When the *InstallCheck* tool is run, it expands the definitions of the classes using the saved parameters, and compares the desired security attributes against the actual file attributes. Any discrepancies are reported. Annotations for definitions or constraints in the policy are displayed when these problems are found.

Thus, the tool provides a way of describing access policy violations at a more abstract level than just checking a particular instantiation.

5.2 *InstallCheck* example

An example illustrating the *InstallCheck* tool is contained in Appendix C.

6 Conclusions

Each of the tools developed for this effort made use of a factored security presentation in order to aid in identifying potential security problems. The System Security Analysis Tool provides a way to combine a security analysis of a generic architecture with the specific requirements of a particular system. The Application System Call Analysis Tool provides a way of identifying potential weaknesses of a trusted application based on limitations, or issues, of the underlying operating system. The Application Installation Analysis tool provides a way of identifying differences between the way an access control policy is implemented and the vendor's policy.

The security factorizations used by the tools reduce the difficulty of tool use by utilizing pre-analyzed parts. In the case of the System Security Analysis Tool, a generic architecture, generic security analysis, and set of residual security issues can be developed prior to use by someone developing a specific system. In the case of application system call analysis, a security product profile can be developed prior to analyzing the security of a trusted application. In the case of installation analysis, the vendor can provide an access security policy that can be instantiated and then checked as needed.

7 Recommendations

We recommend that more security information be made available in order to help system developers, administrators, and certifiers analyze the security of composite systems. This includes:

- generic architectures, including their security characteristics and residual security issues
- product profiles, including security concerns associated with the product interface or system calls.

We also recommend that trusted application policies be designed so that configuration checking can refer to the vendor's application policy and not just an instantiated version of that policy.

8 References

CORE[®] User Reference Guide, Copyright © 1993-96, Vitech Corporation, 2070 Chain Bridge Road, Suite 105, Vienna, Virginia, 22182-2536.

“Final Report on Composability Methodology”, David Rosenthal, Geoffrey Hird, Daryl McCullough, and Roshan Thomas, Odyssey Research Associates, TM-95-0089, 1995.

The HP-UX CMW 10.09.01 Security Reference, Volume 4, Section 2: System Calls. Hewlett Packard Part No. SF239-90621, Fourth Edition.

HP-UX 10.09.01 CMW Trusted Facility Manual, HP Part No. SF238-90622, E0496, Edition 3, Copyright 1996, Hewlett-Packard Company.

HP-UX 10.09.01 CMW Key Security Concepts, HP Part No. SF238-90623, E0496, Edition 3, Copyright 1996, Hewlett-Packard Company.

HP-UX 10.09.01 CMW Security Features User's Guide, HP Part No. SF240-90620, E0496, Edition 3, Copyright 1996, Hewlett-Packard Company.

HP-UX 10.09.01 CMW Security Features Programmer's Guide, HP Part No. SF240-90621, E0496, Edition 3, Copyright 1996, Hewlett-Packard Company.

HP-UX 10.09.01 CMW Trusted Facility Administrator's Reference Manual, HP Part No. SF239-90620, E0496, Edition 3, Copyright 1996, Hewlett-Packard Company.

Managing HP-UX Software with SD-UX, HP 9000 Computers, HP Part No. B2355-90080, E0695, June 1995.

Software User Manual for the System Security Analysis Tool

Software User Manual for the Application System Call Analysis Tool

Software User Manual for the Application Installation Analysis Tool

9 Appendix A: Example Report of System Security Analysis Tool

SECURITY ANALYSIS FOR SatImageSystem

9.1 Introduction

This report describes the security relevant information produced by the security analysis tool. It includes a summary of the system security features, the results of the tool's security analysis, and generic system appendix information.

9.2 System Overview

The Satellite Imagery System downgrades satellite reconnaissance imagery via an encrypted transmission to an SCI level imagery processing and analysis enclave, Enclave1. Imagery undergoes preprocessing, with further imagery analysis being split between Enclave1 and Enclave2, which are at the same sensitivity level and are connected by a two-way secure channel. Images suitable for downgrade are sent from Enclave1 through an ISSE guard to a low side LAN for further dissemination.

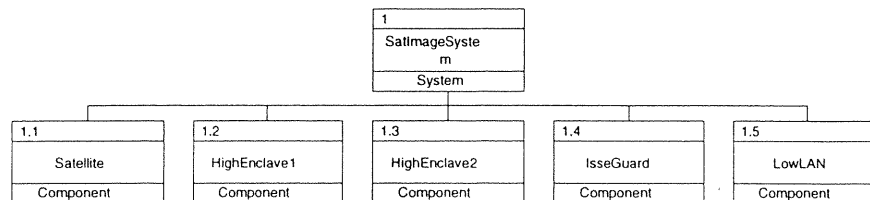


Figure 5 - Physical Hierarchy

9.3 Component Summary

9.3.1 Satellite

The Crystal reconnaissance satellite collects real-time digital imagery for encrypted transmission.

9.3.2 HighEnclave1

Enclave1 is engaged in the processing and analysis of satellite reconnaissance imagery.

9.3.3 HighEnclave2

Enclave2 is engaged in the processing and analysis of satellite reconnaissance imagery.

9.3.4 IsseGuard

The ISSE Guard downgrades imagery from the high side Enclave1 to a low side LAN for dissemination.

9.3.5 LowLAN

The low side LAN receives imagery data from the high side Enclave1 via the ISSE Guard.

9.4 Component Security Summary

9.4.1 Satellite

The satellite is a sealed orbiting device, and is thus considered secure.
[TOP SECRET, TOP SECRET]

9.4.2 HighEnclave1

Access to Enclave1 is both physically and electronically restricted. The enclave itself has a guarded perimeter with a secure checkpoint. Data sent from the enclave to a lower sensitivity level must pass through a guarded downgrade path.
[TOP SECRET/SCI/, TOP SECRET/SCI/]

9.4.3 HighEnclave2

Access to Enclave2 is both physically and electronically restricted. The enclave itself has a guarded perimeter with a secure checkpoint. Data in the enclave may only be sent via a secure channel to Enclave1, which has the same sensitivity level.
[TOP SECRET/SCI/, TOP SECRET/SCI/]

9.4.4 IsseGuard

[SECRET, TOP SECRET]

9.4.5 LowLAN

Access to Low LAN is both physically and electronically restricted. The terminals in the Low LAN are in locked offices and are unconnected to any open networks.
[SENSITIVE, BUT UNCLASSIFIED, SENSITIVE, BUT UNCLASSIFIED]

9.5 Link Security Summary

9.5.1 High1toHigh2

[TOP SECRET, TOP SECRET]

9.5.2 High1toIsseGuard

[TOP SECRET, TOP SECRET]

9.5.3 IsseGuardtoLowLAN

[SECRET, SECRET]

9.5.4 SatellitetoHigh1

[TOP SECRET, TOP SECRET]

9.6 Security Analysis Tests

9.6.1 ComponentLinkLevels [Potential Problem Detected]

9.6.2 GenericToSpecific [Potential Problem Detected]

9.6.3 PhysicalProtection [Potential Problem Detected]

9.7 Component Results

9.7.1 Satellite

9.7.2 HighEnclave1

Test PhysicalProtection has yielded the following result with respect to component element HighEnclave1: Physical protection is unspecified.

Test GenericToSpecific has yielded the following result:

The component HighEnclave1 does not address the generic requirement of generic component HighIsseEnclave.

9.7.3 HighEnclave2

9.7.4 IsseGuard

9.7.5 LowLAN

Test ComponentLinkLevels has yielded the following result with respect to component element LowLAN and link element IsseGuardtoLowLAN: There is a security level mismatch.

The associated MAC levels for the component and link elements are [SENSITIVE, BUT UNCLASSIFIED, SENSITIVE, BUT UNCLASSIFIED] and [SECRET, SECRET], respectively.

9.8 Generic Security Summary: IsseGuardedSystem

9.8.1 Generic Description

The ISSE-type guarded system aids in proper downgrading by providing message integrity and authentication, as well as providing manual review. The guard accepts files for downgrade that have been encapsulated and manually reviewed at high. The guard recomputes the CRC to ensure data integrity, as well as validating source/destination

addresses and message security classification. The guard also accepts files for upgrade, after checking source/destination addresses.

9.8.2 Generic Security Assessment

Most of the security of the ISSE-guarded system is enforced in the high enclave rather than in the guard. Message integrity is enforced via encapsulation, while message appropriateness for downgrade is enforced via manual review. Both of these activities occur before the message reaches the guard. The role of the guard is simply to confirm message integrity through recalculation of the CRC checksum. In addition, firewall protection must be provided by the low enclave, as the guard does not possess such protection.

The ISSE-type guarded system prevents message modification between formation and downgrade by providing message encapsulation. The system also prevents inappropriate downgrade by manually reviewing message content. The low side enclave of the system is a protected enclave, eliminating the need for firewall protection at the guard.

The ISSE Guard has been predominantly fielded with an SCI High enclave and a Secret NOFORN Low enclave.

9.8.3 Generic Requirements and Specific Issues

9.8.3.1 ManualReview

Requirement:

Manual review at high should be carried out in such a way as to minimize the risk of inappropriate downgrade.

Response:

Following CRC encapsulation, a two-person read-only manual review is carried out. Two-person review provides an additional check of message downgrade appropriateness, while review in read-only mode protects against message rejection by the guard due to reviewer modification or insertion.

9.8.3.2 MessageFormation

Requirement:

Messages must be protected from modification prior to encapsulation. Such protection may be based upon considerations such as the trustworthiness of the drafting software, the trustworthiness of the path between message formation and message encapsulation, or the trustworthiness of the high enclave in general.

Response:

Imagery is encrypted and CRC encapsulated in the satellite prior to transmission. Since the satellite is considered to be secure, images are considered to be protected from modification prior to encapsulation.

9.8.3.3 MessageFormatting

Requirement:

Messages formed at high must in general be limited to a format which may be fully manually reviewed, in order to minimize the possible exploitation of covert channels. If another format is used, additional precautions must be taken to prevent inappropriate downgrade.

Response:

Since imagery is generated and encapsulated in the satellite, it is considered to be secure from possible covert channel insertion during formation. Thus, although imagery in general is susceptible to covert channel exploitation, the satellite generated imagery is considered secure from inappropriate downgrade through such means.

9.8.3.4 FirewallProtection

Requirement:

The low side enclave should be protected, either by a firewall or some other means, to minimize exposure to possible outside attack.

Response:

The low side is a protected secret enclave unconnected to any open network, therefore it has minimal exposure to outside attack.

10 Appendix B: Example of Application System Call Analysis Tool

The inputs of the ASCA Tool are a description file, specifying various security related descriptions for the system calls, and a filter file, specifying system calls to ignore during the running of the tool. Below, we show examples of both a truncated description file and a filter file. We then show the command-line execution of the tool and the corresponding truncated output. In the particular example we have given, we have run the tool on a library archive. The first section of the output shows all the archive files, listing the found system calls under each file name. The next section of the output contains the description field associated with each flagged system call.

10.1 ASCA description file: warnings.dsc

#ASCA DESCRIPTION FILE

#This asca tool description file has been generated by

#including the following HP-UX man page header fields:

#WARNINGS

SYSTEM CALL alarm

In some implementations, error bounds for alarm are -1, +0 seconds (for the posting of the alarm, not the restart of the process). Thus a delay of 1 second can return immediately. The setitimer() routine can be used to create a more precise delay.

SYSTEM CALL chown

The default operation of chown and chgrp for symbolic links has changed as of HP-UX release 10.0. Use the -h option to get the former default operation.

SYSTEM CALL ioctl

Check all references to signal(5) for appropriateness on systems that support sigvector(2). sigvector(2) can affect the behavior described on this page.

10.2 ASCA filter file: filter_file.flt

#ASCA FILTER FILE

execle

fork

getrlimit

lseek

semop

setgid

setuid

signal

umask

kill

unlink

10.3 ASCA command-line execution

setenv ASCA_PATH ./samples

./asca_tool -d warnings.dsc -f filter_file.flt /lib/libsecurity.a > asca_out

10.4 ASCA output: asca_out

Description File: ./samples/warnings.dsc

Filter File: ./samples/filter_file.flt

/lib/libsecurity.a[accept_pw.o]:

/lib/libsecurity.a[acllib.o]:

chown

```

        ioctl

/lib/libsecurity.a[auditdb.o]:
        .
        .
        .
/lib/libsecurity.a[getprdfent.o]:

/lib/libsecurity.a[identity.o]:
        alarm

/lib/libsecurity.a[lchilabel.o]:
        .
        .
        .

```

____SYSTEM CALL DESCRIPTIONS____

SYSTEM CALL alarm

In some implementations, error bounds for alarm are -1, +0 seconds (for the posting of the alarm, not the restart of the process). Thus a delay of 1 second can return immediately. The setitimer() routine can be used to create a more precise delay.

SYSTEM CALL chown

The default operation of chown and chgrp for symbolic links has changed as of HP-UX release 10.0. Use the -h option to get the former default operation.

SYSTEM CALL ioctl

Check all references to signal(5) for appropriateness on systems that support sigvector(2). sigvector(2) can affect the behavior described on this page.

11 Appendix C: Example of *InstallCheck*

Below we provide a simple example showing the sample inputs and outputs.

The set of inputs comprises:

- a sample security specification file,
- the corresponding security parameter file, and
- the file attribute list.

The set of outputs comprises:

- A report indicating the set of discrepancies between the vendor supplied policy and the current state of the application.

Sample Security Specification file

```

USER_CLASS Data_Administrator
  DESCRIPTION: `The user trusted to maintain the sensitive data`
  CLASS_DEFINITION: $Trusted_Admin
  ATTRIBUTE_DEFINITIONS:
    COMMAND_AUTHS
    NOTE `executables need allowdacread privilege to read sensitive
data`
  ATTRIBUTE_CONSTRAINTS:
    INCLUDES BASE_PRIVILEGES chown

OBJECT_CLASS Log_Dir
  DESCRIPTION: `Multi-level Log Directory`
  CLASS_DEFINITION: $LogDir
  ATTRIBUTE_DEFINITIONS:
    TYPE m
  ATTRIBUTE_CONSTRAINTS:
    SLEVEL INRANGE [confidential, syshi]
    NOTE `No unclassified log records should be generated`

OBJECT_CLASS Bin_Dir
  DESCRIPTION: `Directory containing binaries.`
  CLASS_DEFINITION: $BinDir
  ATTRIBUTE_DEFINITIONS:
    OWNER isso
  ATTRIBUTE_CONSTRAINTS:
    INITIAL_PART_ACL <@.®,rwx>, <*.®,rx>

OBJECT_CLASS Trusted_Executable
  DESCRIPTION: `The analysis tools`
  CLASS_DEFINITION: $BinDir/analyzedata1 , $BinDir/analyzedata2
  ATTRIBUTE_DEFINITIONS:
    IS_EXECUTABLE TRUE
    MODE rwxr-xr-x

OBJECT_CLASS Trusted_Update_Tools
  DESCRIPTION: `The maintenance tools`
  CLASS_DEFINITION: $BinDir/updatedata
  ATTRIBUTE_DEFINITIONS:
    IS_EXECUTABLE TRUE
    OWNER CLASS_NAME Data_Administrator
    PPRIVS allowdacwrite ,allowdacread

```

Corresponding Sample Security Parameter file

Fill in missing values (marked by ????) with your system-specific values

```
$Trusted_Admin david
$SourceDir /task2/BTool/example/src
$BinDir /task2/BTool/example/bin
$LogDir /task2/BTool/example/log2
$AnalysisGroup sys
```

Corresponding File Attribute List

```
task2/BTool/example/log2:\
    :f_owner=root:f_group=sys:f_mode#0755:f_type=d:\
    :f_slevel=syslo:\
    :chkent:
/task2/BTool/example/log2/file1:\
    :f_owner=root:f_group=sys:f_mode#0440:f_type=r:\
    :f_slevel=CONFIDENTIAL:\
    :chkent:
/task2/BTool/example/log2/file2:\
    :f_owner=root:f_group=sys:f_mode#0440:f_type=r:\
    :f_slevel=SECRET:\
    :chkent:
/task2/BTool/example/log2/file3:\
    :f_owner=root:f_group=sys:f_mode#0440:f_type=r:\
    :f_slevel=SECRET:\
    :chkent:
/task2/BTool/example/bin:\
    :f_owner=root:f_group=other:f_mode#0755:f_type=d:\
    :f_pprivs=suspendaudit, allowdacread:\
    :f_acl= <@.@,r> <@.@,all> <*.*,-> <@.*,rx>:\
    :f_slevel=syslo:\
    :chkent:
/task2/BTool/example/bin/analyzedata1:\
    :f_owner=root:f_group=sys:f_mode#0550:f_type=r:\
    :f_pprivs=suspendaudit, allowdacread:\
    :f_slevel=SECRET:\
    :chkent:
/task2/BTool/example/bin/analyzedata2:\
    :f_owner=root:f_group=sys:f_mode#0550:f_type=r:\
    :f_pprivs= allowdacread:\
    :f_slevel=SECRET:\
    :chkent:
/task2/BTool/example/bin/updatedata:\
    :f_owner=root:f_group=sys:f_mode#0550:f_type=r:\
    :f_pprivs=allowdacwrite, allowdacread:\
    :f_slevel=SECRET:\
    :chkent:
```

The sample output generated when the *InstallCheck* tool is executed with the above set of inputs

```
User has bypassdac command authorization
User Account Name: root
*****
*****
Checking Obj Class: Log_Dir
-----
File Name : /task2/BTool/example/log2
-----
-> ERROR: The type should be m, but is d
-> ERROR: The sensitivity level is not between confidential and syshi
->     NOTES: No unclassified log records should be generated
*****
Checking Obj Class: Bin_Dir
-----
File Name : /task2/BTool/example/bin
-----
-> ERROR: The owner should be isso, but is root
-> ERROR: ACL <*.*,rx> was not found in the FAL
*****
Checking Obj Class: Trusted_Executable
-----
File Name : /task2/BTool/example/bin/analyzedata1
-----
-> ERROR: The mode should be 755(octal), but is 550(octal)
-----
File Name : /task2/BTool/example/bin/analyzedata2
-----
-> ERROR: The mode should be 755(octal), but is 550(octal)
*****
Checking Obj Class: Trusted_Update_Tools
-----
File Name : /task2/BTool/example/bin/updatedata
-----
-> ERROR: The specified owner, david, of the file
/task2/BTool/example/bin/updatedata not found in the system
*****
End of InstallCheck Program
```

12 List of Symbols

ACSA -- Application System Call Analysis
HP-CMW -- HP Compartmented Mode Workstation